Sujet Gomoku

m billaud

18/03/2021

Table des matières

1 Le Jeu de Gomoku (Morpion) 1

2 Travail demandé 1
2.1 Programme 1
2.2 Dossier 1

3 Progression du travail (exemple) 1

1 Le Jeu de Gomoku (Morpion)

Jeu à deux joueurs (Blanc et Noir), qui posent des rôle des pions sur un plateau de grande taille (par exemple 20 sur 20). Les règles :

- Les joueurs posent des pions à tour de rôle dans une case vide.
- On doit jouer à côté d'une case déjà occupée, sauf au premier coup.
- Le gagnant est le premier qui aligne 5 pions selon une ligne, colonne, ou diagonale.
- la partie est nulle si toutes les cases sont occupées.

2 Travail demandé

2.1 Programme

Écrire un programme Java permettant de mener des parties à deux joueurs (humains ou machines).

Ce programme sera conçu en mettant à profit la programmation orientée-objets, en vous attachant particulierement à bien attribuer les responsabilités entre des objets, par exemple :

- le plateau, qui représente un tableau de cases remplies ou non ;
- la partie, qui représente l'état du jeu,
- un match, qui fait dérouler une partie entre deux joueurs,
- les joueurs, qui indiquent le coup qu'ils choisissent d'effectuer à chaque moment de la partie.

2.2 Dossier

Outre le programme, soigneusement commenté, vous remettrez un dossier dans lequel vous préciserez

- la répartition des responsabilités entre les classes.
- les choix effectués pour la programmation.

---

3 Progression du travail (exemple)

Ci-dessous on présente une progression possible pour arriver à réaliser ce projet par étapes.

1. Partir de l'énumération :

```
enum Color {BLACK,  WHITE,  NONE};
```

qui servira à décrire le contenu des cases (et aussi désigner le joueur qui doit jouer).

2. Les positions (numéro de ligne et de colonne) sont contenues dans des instances de `Position`

```
class Position {
   final int row, col;
}
```

Elle peuvent être définies à partir d'une paire d'entiers (`new Position (0, 3)`) ou une chaine `new Position("A4")`. On utilise une lettre pour la colonne A=0, B=1, etc. et un nombre pour la colonne (attention 1 correspond à la colonne 0). L'appel du constructeur avec des coordonnées invalides lève une exception (`InvalidCoordinates`)

Il faudra écrire une classe de tests unitaires pour en vérifier le bon fonctionnement, en particulier que "A14" correspond bien à (0, 13). et que "J0" lève une exception.

3. Dans la classe `Board`, des méthodes permettent de remplir/consulter les cases.

```
void set(Position p, Color c);
Color get(Position p);
```

4. Une classe `Game` représente l'état du jeu.

- un attribut `Color nextPlayer;` indique le joueur qui doit jouer le prochain coup.
- un attribut `Board board` contient l'état du plateau,
- on conserve la liste des coups joués dans un tableau (ArrayList).
- une méthode `bool play(Position p)` tente d'exécuter un coup du joueur au trait à la position indiquée. Si c'est possible, le plateau est modifié, le joueur au trait change, et la méthode retourne `true`.
- des indicateurs consultables sont tenus à jour pour savoir si la partie est finie, et son résultat.

5. Une classe `Match` fait dérouler une partie entre deux `Players` sur un `Board`.

```
interface Player {
    Position choice(Board b);
}
```

Elle les fait jouer tour à tour, en leur présentant l'état du plateau et en leur demandant de choisir un coup. Un matche est lancé ainsi :

```
Match m = new Match(20, 20,
    new HumanPlayer("Anna"),
    new HumanPlayer("Bob"));
m.run();
```

Un `HumanPlayer` dialogue en mode texte. Il affiche le contenu de la grille sur l'écran, et interroge l'utilisateur sur le coup qu'il veut jouer. Maquette de taille réduite 5 x 8 :

```
    A B C D E F G H
  +-----------------+
5 |                 |
4 |     O   X O     |
3 |     X O O       |
2 |         X       |
1 |                 |
  +-----------------+

Anna plays X at :
```

Important : le `Player` n'est pas chargé d'effectuer le coup. C'est le `Game` qui le fera, ou réinterrogera le `Player` si le coup n'est pas possible. (La méthode `Game ::play` sera privée).

6. Un joueur peut mettre fin à la partie en tapant "`/quit`". Ce qui lève une exception `GamePlayerLeaves`.

7. Vous ferez affronter un joueur humain et un `RobotPlayer`, une intelligence artificielle qui joue automatiquement (à un endroit quelconque autorisé, ça fera l'affaire). Le `RobotPlayer` n'affiche rien.

8. En fin de partie, le `Game` affiche la liste de coups joués, et le résultat.